



INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification ⁶ : H04B		A2	(11) International Publication Number: WO 99/55013
			(43) International Publication Date: 28 October 1999 (28.10.99)
(21) International Application Number: PCT/US99/08673			(81) Designated States: AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, UZ, VN, YU, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SL, SZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).
(22) International Filing Date: 20 April 1999 (20.04.99)			
(30) Priority Data: 09/063,492 20 April 1998 (20.04.98) US			Published <i>Without international search report and to be republished upon receipt of that report.</i>
(71) Applicant: SUN MICROSYSTEMS, INC. [US/US]; 901 San Antonio Road, M/S UPAL01-521, Palo Alto, CA 94303 (US).			
(72) Inventors: RUBERG, Alan, T.; 605 Emerald Bay Lane, Foster City, CA 94404 (US). HANKO, James, G.; 2746 Ohio Avenue, Redwood City, CA 94061 (US). NORTHCUTT, J., Duane; 184 Seminary Drive, Menlo Park, CA 94025 (US). WALL, Gerard, A.; 4515 Crocus Drive, San Jose, CA 95136 (US).			
(74) Agents: HECKER, Gary, A. et al.; Hecker & Harriman, Suite 2300, 1925 Century Park East, Los Angeles, CA 90067 (US).			
(54) Title: METHOD AND APPARATUS OF SUPPORTING A VIDEO PROTOCOL IN A NETWORK ENVIRONMENT			
(57) Abstract			
<p>A method and apparatus of supporting a video protocol in a network environment. In an embodiment of the invention, video processing and hardware requirements associated with a receiver are minimized by specifying a single video protocol for transmission of video data between transmitters and receivers on a network. The protocol specifies a color format that allows for high video quality and minimizes the complexity of the receiver. Transmitters are equipped with transformation mechanisms that provide for conversion of video data into the designated protocol as needed. Compression of the components of the color format is provided to reduce transmission bandwidth requirements. In one embodiment, aspects of the designated protocol compensate for problems associated with transmitting video data over a network. The designated protocol specifies a color format including a luminance value and two chrominance values. Quantized differential coding is applied to the luminance value and subsampling is performed on the chrominance values to reduce transmission bandwidth requirements. Upscaling of video data is performed at the receiver, whereas downscaling is performed at the transmitter. Various display sizes can thus be accommodated with efficient use of network bandwidth.</p>			
<pre> graph TD 600[ACQUIRE VIDEO DATA] --> 601[DECOMPRESS OR CONVERT DATA TO YUV] 601 --> 602[DOWNSCALE IF NEEDED] 602 --> 603[COMPRESS LUMA VALUES INTO DATA PACKET] 603 --> 604[SUBSAMPLE CHROMA VALUES AND LOAD INTO DATA PACKET] 604 --> 605[TRANSMIT DATA PACKET] </pre>			

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	ES	Spain	LS	Lesotho	SI	Slovenia
AM	Armenia	FI	Finland	LT	Lithuania	SK	Slovakia
AT	Austria	FR	France	LU	Luxembourg	SN	Senegal
AU	Australia	GA	Gabon	LV	Latvia	SZ	Swaziland
AZ	Azerbaijan	GB	United Kingdom	MC	Monaco	TD	Chad
BA	Bosnia and Herzegovina	GE	Georgia	MD	Republic of Moldova	TG	Togo
BB	Barbados	GH	Ghana	MG	Madagascar	TJ	Tajikistan
BE	Belgium	GN	Guinea	MK	The former Yugoslav Republic of Macedonia	TM	Turkmenistan
BF	Burkina Faso	GR	Greece	ML	Mali	TR	Turkey
BG	Bulgaria	HU	Hungary	MN	Mongolia	TT	Trinidad and Tobago
BJ	Benin	IE	Ireland	MR	Mauritania	UA	Ukraine
BR	Brazil	IL	Israel	MW	Malawi	UG	Uganda
BY	Belarus	IS	Iceland	MX	Mexico	US	United States of America
CA	Canada	IT	Italy	NE	Niger	UZ	Uzbekistan
CF	Central African Republic	JP	Japan	NL	Netherlands	VN	Viet Nam
CG	Congo	KE	Kenya	NO	Norway	YU	Yugoslavia
CH	Switzerland	KG	Kyrgyzstan	NZ	New Zealand	ZW	Zimbabwe
CI	Côte d'Ivoire	KP	Democratic People's Republic of Korea	PL	Poland		
CM	Cameroon			PT	Portugal		
CN	China	KR	Republic of Korea	RO	Romania		
CU	Cuba	KZ	Kazakhstan	RU	Russian Federation		
CZ	Czech Republic	LC	Saint Lucia	SD	Sudan		
DE	Germany	LI	Liechtenstein	SE	Sweden		
DK	Denmark	LK	Sri Lanka	SG	Singapore		
EE	Estonia	LR	Liberia				

METHOD AND APPARATUS OF SUPPORTING A VIDEO PROTOCOL IN A NETWORK ENVIRONMENT

5

1. FIELD OF THE INVENTION

This invention relates to the field of digital video, and, more specifically, to digital video applications in a network environment.

10

2. BACKGROUND ART

Computers and computer networks are used to exchange information in many fields such as media, commerce, and telecommunications, for 15 example. One form of information that is commonly exchanged is video data (or image data), i.e., data representing a digitized image or sequence of images. A video conferencing feed is an example of telecommunication information which includes video data. Other examples of video data include video streams or files associated with scanned images, digitized television 20 performances, and animation sequences, or portions thereof, as well as other forms of visual information that are displayed on a display device. It is also possible to synthesize video information by artificially rendering video data from two or three-dimensional computer models.

25 For the purposes of this discussion, the exchange of information between computers on a network occurs between a "transmitter" and a "receiver." In video applications, the information contains video data, and the services provided by the transmitter are associated with the processing

and transmission of the video data. A problem with current network systems is that multiple services provided by one or more transmitters may provide video data using different video protocols. The complexity of the receiver is necessarily increased by the need to accommodate each of the different video 5 protocols. Also, the amount of data associated with video applications is very large. The transmission of such large amounts of data over a network can result in bandwidth utilization concerns. The following description of video technology and an example network scheme are given below to provide a better understanding of the problems involved in transmitting video data 10 over a network.

General Video Technology

In digital video technology, a display is comprised of a two 15 dimensional array of picture elements, or "pixels," which form a viewing plane. Each pixel has associated visual characteristics that determine how a pixel appears to a viewer. These visual characteristics may be limited to the perceived brightness, or "luminance," for monochrome displays, or the visual characteristics may include color, or "chrominance," information. 20 Video data is commonly provided as a set of data values mapped to an array of pixels. The set of data values specify the visual characteristics for those pixels that result in the display of a desired image. A variety of color models exist for representing the visual characteristics of a pixel as one or more data values.

25

RGB color is a commonly used color model for display systems. RGB color is based on a "color model" system. A color model allows convenient specification of colors within a color range, such as the RGB (red, green, blue)

primary colors. A color model is a specification of a three-dimensional color coordinate system and a three-dimensional subspace or "color space" in the coordinate system within which each displayable color is represented by a point in space. Typically, computer and graphic display systems are three-

5 phosphor systems with a red, green and blue phosphor at each pixel location.

The intensities of the red, green and blue phosphors are varied so that the combination of the three primary colors results in a desired output color.

The RGB color model uses a Cartesian coordinate system. The

10 subspace of interest in this coordinate system is known as the "RGB color cube" and is illustrated in Figure 1. Each corner of the cube represents a color that is theoretically one-hundred percent pure — that is, the color at that location contains only the color specified, and contains no amount of other colors. In the RGB color cube, the corners are defined to be black, white, red,

15 green, blue, magenta, cyan, and yellow. Red, green and blue are the primary colors, black is the absence of color, white is the combination of all colors, and cyan, magenta and yellow are the complements of red, green and blue.

Still referring to Figure 1, the origin of the coordinate system

20 corresponds to the black corner of the color cube. The cube is a unit cube so that the distance between the origin and adjacent corners is 1. The red corner is thus at location, (1,0,0). The axis between the origin (black) and the red corner is referred to as the red axis 110.

25 The green corner is at location (0,1,0) and the axis 120 between the black origin and the green corner is referred to as the green axis. The blue corner is at location (0,0,1) and the blue axis 130 is the axis between the blue corner and the origin.

Cyan is at corner (0,1,1), magenta is at corner (1,0,1) and yellow is at corner (1,1,0). The corner opposite the origin on the cube's diagonal at location (1,1,1) is the white corner.

5

A color is defined in the color cube by a vector having red, green and blue components. For example, vector 180 is the resultant of vectors 180R, (along the red axis), vector 180G (along the green axis) and vector 180B (along the blue axis). The end point of vector 180 can be described mathematically by 10 0.25R + 0.50G + 0.75B. The end of this vector defines a point in color space represented mathematically by the sum of its red, green and blue components.

An example of a system for displaying RGB color is illustrated in 15 Figure 2. A refresh buffer 140, also known as a video RAM, or VRAM, is used to store color information for each pixel on a video display, such as CRT display 160. A DRAM can also be used as buffer 140. The VRAM 140 contains one memory location for each pixel location on the display 160. For example, pixel 190 at screen location X_0Y_0 corresponds to memory location 150 in the 20 VRAM 140. The number of bits stored at each memory location for each display pixel varies depending on the amount of color resolution required. For example, for word processing applications or display of text, two intensity values are acceptable so that only a single bit need be stored at each memory location (since the screen pixel is either "on" or "off"). For color images, 25 however, a plurality of intensities must be definable. For certain high end color graphics applications, it has been found that twenty-four bits per pixel produces acceptable images.

Consider, for example, that in the system of Figure 2, twenty-four bits are stored for each display pixel. At memory location 150, there are then eight bits each for the red, green and blue components of the display pixel. The eight most significant bits of the VRAM memory location could be used to

5 represent the red value, the next eight bits represent the green value and the eight least significant bits represent the blue value. Thus, 256 shades each of red, green and blue can be defined in a twenty-four bit per pixel system.

When displaying the pixel at X0, Y0, the bit values at memory location 150 are provided to video driver 170. The bits corresponding to the R component are

10 provided to the R driver, the bits representing the green component are provided to the G driver, and the bits representing the blue component are provided to the blue driver. These drivers activate the red, green and blue phosphors at the pixel location 190. The bit values for each color, red, green and blue, determine the intensity of that color in the display pixel. By

15 varying the intensities of the red, green and blue components, different colors may be produced at that pixel location.

Color information may be represented by color models other than RGB. One such color space is known as the YUV (or Y'CbCr as specified in

20 ITU.BT-601) color space which is used in the commercial color TV broadcasting system. The YUV color space is a recoding of the RGB color space, and can be mapped into the RGB color cube. The RGB to YUV conversion that performs the mapping is defined by the following matrix equation:

$$\begin{bmatrix} Y' \\ U' \\ V' \end{bmatrix} = \begin{bmatrix} Y' \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.169 & -0.331 & 0.500 \\ 0.500 & -0.419 & -0.081 \end{bmatrix} \cdot \begin{bmatrix} R' \\ G' \\ B' \end{bmatrix}$$

The inverse of the matrix is used for the reverse conversion. The Y axis of the YUV color model represents the luminance of the display pixel, 5 and matches the luminosity response curve for the human eye. U and V are chrominance values. In a monochrome receiver, only the Y value is used. In a color receiver, all three axes are used to provide display information.

In operation, an image may be recorded with a color camera, which is 10 an RGB system, and converted to YUV for transmission. At the receiver, the YUV information is then retransformed into RGB information to drive the color display.

Many other color models are also used to represent video data. For 15 example, CMY (cyan, magenta, yellow) is a color model based on the complements of the RGB components. There are also a variety of color models, similar to YUV, which specify a luminance value and multiple chrominance values, such as the YIQ color model. Each color model has its own color transformation for converting to a common displayable video 20 format such as RGB. Most transformations may be defined with a transform matrix similar to that of the YIQ color space.

There are many color formats used in the prior art for transmitting image and video data over networks. Some examples of existing color

formats are H.261 and H.263, which are used in digital telephony, and MPEG1, MPEG2 and MJPEG. These color formats use compression schemes to reduce the amount of data being transmitted. For example, many color formats use a variation of DCT (discrete cosine transform) compression to perform

5 compression in the frequency domain. A form of variable length Huffman encoding may also be implemented to reduce bandwidth requirements. Specialized compression/decompression hardware or software is often used to perform the non-trivial conversion of data in these color formats to data for display.

10

Network Transmission Of Video Data

As has been described, there exist a variety of color formats for video data. These variations allow for a large number of different possible video

15 protocols. It becomes problematic for a receiver on a network to handle all possible video protocols that might be used by different transmitters and services acting as video data sources on the network. The problems associated with multiple video protocols are described below with reference to the sample network system illustrated in Figure 3. Figure 3 illustrates a

20 sample network system comprising multiple transmitters 300A-300C for sourcing video data and a single receiver 303. Receiver 303 is equipped with one or more display devices for providing video output associated with received video data.

25 In the example of Figure 3, transmitters 300A, 300B and 300C, and receiver 303 are coupled together via network 302, which may be, for example, a local area network (LAN). Transmitter 300A transmits video data along network connection 301A to network 302 using video protocol A.

Transmitter 300B transmits video data along network connection 301B to network 302 using video protocol B. Transmitter 300C transmits video data along network connection 301C to network 302 using video protocol C. Thus, receiver 303 may receive video data over network connection 305 from 5 network 302 under any of video protocols A, B or C, as well as any other protocols used by other transmitters connected to network 302, or used by multiple services embodied within one of transmitters 300A-300C.

Receiver 303 may be equipped with different video cards (i.e., 10 specialized hardware for video processing) or software plug-ins to support each video protocol, but this increases the complexity of the receiver, and necessitates hardware or software upgrades when new video protocols are developed. For systems wherein it is a goal to minimize processing and hardware requirements for a receiver, the added complexity of supporting 15 multiple protocols is undesirable.

An issue in all network applications is utilization of network bandwidth. For video applications, bandwidth is an even greater concern due to the large amounts of data involved in transmitting one or more frames of 20 video data. For example, consider a raw workstation video signal of twenty-four bit RGB data, sent in frames of 1280 x 1024 pixels at thirty frames per second. The raw workstation video represents 240 MBps (megabytes per second) of continuous data. Even for smaller frame sizes, video data can represent a significant load on a network, resulting in poor video 25 performance if the required bandwidth cannot be provided. Further, other applications on the network may suffer as bandwidth allocations are reduced to support the video transmission.

SUMMARY OF THE INVENTION

A method and apparatus of supporting a video protocol in a network environment is described. In an embodiment of the invention, video processing and hardware requirements associated with a receiver are minimized by specifying a single video protocol for transmission of video data between transmitters and receivers on a network. The protocol specifies a color format that allows for high video quality and minimizes the complexity of the receiver. Transmitters are equipped with transformation mechanisms that provide for conversion of video data into the designated protocol as needed. Compression of the components of the color format is provided to reduce transmission bandwidth requirements.

In one embodiment of the invention, aspects of the designated protocol compensate for problems associated with transmitting video data over a network. The designated protocol specifies a color format including a luminance value and two chrominance values. Quantized differential coding is applied to the luminance value and subsampling is performed on the chrominance values to reduce transmission bandwidth requirements. In one embodiment of the invention, upscaling of video data is performed at the receiver, whereas downscaling is performed at the transmitter. Various display sizes can thus be accommodated with efficient use of network bandwidth.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 is a diagram of an RGB color space.

5 Figure 2 is a block diagram of a video display apparatus.

Figure 3 is a block diagram of a network system having a single receiver and multiple transmitters.

10 Figure 4A is a flow diagram illustrating a luma compression scheme in accordance with an embodiment of the invention.

Figure 4B is a flow diagram illustrating a luma decompression scheme in accordance with an embodiment of the invention.

15 Figure 5 is a diagram illustrating a subsampling and upsampling process in accordance with an embodiment of the invention.

Figure 6A is a flow diagram illustrating a video data transmission process in accordance with an embodiment of the invention.

20 Figure 6B is a flow diagram illustrating a video data reception process in accordance with an embodiment of the invention.

25 Figure 7 is a block diagram of a computer execution environment.

Figure 8 is a block diagram of a human interface device computer system.

30 Figure 9 is a block diagram of an embodiment of a human interface device.

DETAILED DESCRIPTION OF THE INVENTION

The invention is a method and apparatus of supporting a video protocol in a network environment. In the following description, numerous specific details are set forth to provide a more thorough description of embodiments of the invention. It will be apparent, however, to one skilled in the art, that the invention may be practiced without these specific details. In other instances, well known features have not been described in detail so as not to obscure the invention.

10

Single Video Protocol For Networked Transmissions

In an embodiment of the invention, a single video protocol is used for transmission of video data between a transmitter and a receiver. The transmitter of the video data is responsible for supplying video data in accordance with the designated protocol. For example, a transmitter and its internal video services are configured to perform any necessary protocol transformations to bring video data into conformance with the designated protocol before transmission to a receiver. Hardware and processing requirements of the receiver are minimized as only one video protocol need be supported at the receiver.

Though discussed in this specification primarily as being applied to video data transmitted one-way from a transmitter to a receiver, video data may also be transmitted from the receiver to the transmitter using the designated protocol. The transmitter may then process the video data in the form of the designated protocol, or the transmitter may convert the video data into another video protocol for further processing.

The designated protocol is chosen to give high video quality to satisfactorily encompass all other video protocols while permitting strategic compression based on knowledge of human perception of luminance and 5 chrominance. The high video quality of the designated protocol ensures that any necessary protocol transformations by a transmitter do not result in a significant loss of video quality from the original video data. An example of a protocol that provides high video quality with compression is a protocol specifying a color format with quantized differential coding of the luminance 10 value and subsampling of chrominance values.

A transmitter may support video applications using the designated protocol, and the transmitter may be configured with mechanisms, such as hardware cards or software plug-ins or drivers, to convert between other 15 video protocols and the designated protocol, for example, using color model matrix transformations.

In an embodiment of the invention, data packets are used to transmit variably sized blocks of video data between a transmitter and a receiver using 20 a connectionless datagram scheme. A connectionless scheme means that each packet of video data, i.e., each video block, is processed as an independent unit, and the loss of a data packet does not affect the processing of other data packets. This independence provides for robust video processing even on unreliable networks where packet loss may be 25 commonplace.

Some networks are prone to periodic packet loss, i.e., packet loss at regular intervals. This periodic behavior can result in the stagnation of

portions of the video display as the same video blocks are repeatedly lost. To prevent video block stagnation, the spatial order in which video blocks are sent to the receiver for display may be pseudo-randomly determined to disrupt any periodicity in packet performance.

5

In one embodiment, the data packets containing video data are provided with a sequence number. By tracking the sequence numbers, the receiver can note when a sequence number is skipped, indicating that the packet was lost during transmission. The receiver can then return to the 10 transmitter a list or range of sequence numbers identifying the lost packet or packets. When the transmitter receives the list or range of sequences, the transmitter can decide whether to ignore the missed packets, resend the missed packets (such as for still images), or send updated packets (such as for streaming video that may have changed since the packet was lost).

15

In one embodiment of the invention, the video data packet comprises the following information:

20

Sequence number - A video stream is processed as a series of blocks of video data. The sequence number provides a mechanism for the receiver to tell the transmitter what sequence numbers have been missed (e.g., due to packet loss), so that the transmitter may determine whether to resend, update or ignore the associated video block.

25

X field - The X field designates the x-coordinate of the receiver's display device wherein the first pixel of the video block is to be displayed.

30

Y field - The Y field designates the y-coordinate of the receiver's display device wherein the first pixel of the video block is to be displayed.

Width - The width field specifies the width of the destination rectangle on the receiver's display device wherein the video block is to be displayed.

5 Height - The height field specifies the height of the destination rectangle on the receiver's display device wherein the video block is to be displayed.

10 Source_w - The source width specifies the width of the video block in pixels. Note that the source width may be smaller than the width of the destination rectangle on the receiver's display device. If this is so, the receiver will upscale the video block horizontally to fill the width of the destination rectangle. The source width should not be larger than the width of the destination rectangle as this implies downscaling, which should be performed by the transmitter for efficiency.

15 Source_h - The source height specifies the height of the video block in pixels. Note that, as with source_w, the source height may be smaller than the height of the destination rectangle on the receiver's display device. As above, the receiver will upscale the video block vertically to fill the height of the destination rectangle. The source height should not be larger than the height of the destination rectangle as this implies downscaling, which should be performed by the transmitter for efficiency.

20 Luma encoding - The luma encoding field allows the transmitter to designate a particular luma encoding scheme from a set of specified luma encoding schemes.

25 Chroma_sub_X - This field allows the transmitter to designate the degree of horizontal subsampling performed on the video data chroma values.

30 Chroma_sub_Y - This field allows the transmitter to designate the degree of vertical subsampling performed on the video data chroma values.

35 Video data - The video data includes (source_w * source_h) pixel luma values (Y), and ((source_w/chroma_sub_x) * (source_h/chroma_sub_y)) signed chroma values (U, V or Cb, Cr).

YUV (Y'CbCr) Color

In an embodiment of the invention, the color model of the chosen protocol is specified by the International Telecommunications Union in 5 ITU.BT-601 referring to an international standard for digital coding of television pictures using video data components Y'CbCr, where Y' is a luminance or "luma" value, Cb (or U') is a first chromaticity or "chroma" value represented as a blue color difference proportional to (B'-Y') and Cr (or V') is a second chroma value represented as a red color difference 10 proportional to (R'-Y') (Note that primed values such as Y' indicate a gamma corrected value). This ITU specification is independent of any scanning standard and makes no assumptions regarding the "white" point or CRT gamma. For $0 \leq (R, G, B) \leq 1$, the range for Y' is $0 \leq Y' \leq 1$ and the range for Cb and Cr is $-0.5 \leq (Cb, Cr) \leq 0.5$.

15

The R'G'B' \leftrightarrow Y'CbCr color transforms are as follows:

$$\begin{bmatrix} Y' \\ U' \\ V' \end{bmatrix} = \begin{bmatrix} Y' \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.169 & -0.331 & 0.500 \\ 0.500 & -0.419 & -0.081 \end{bmatrix} \cdot \begin{bmatrix} R' \\ G' \\ B' \end{bmatrix}$$

$$\begin{bmatrix} R' \\ G' \\ B' \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1.403 \\ 1 & -0.344 & -0.714 \\ 1 & 1.773 & 0 \end{bmatrix} \cdot \begin{bmatrix} Y' \\ U' \\ V' \end{bmatrix}$$

20

Under the specified protocol, the transmitter performs any transformations required to convert the video data into the YUV format. This may include performing the RGB to YUV matrix conversion shown above to convert RGB data. Transformations may also include

5 decompression from other color formats (e.g., H.261, MPEG1, etc.). The receiver can drive an RGB display device by performing the above matrix operation to convert incoming YUV (Y'CbCr) data received from a transmitter into RGB data for display at the display rectangle identified in the data packet. No other color transformations are necessary at the receiver.

10 The receiver is also able to accept RGB data in the same video block format because RGB data is directly supported in the receiver. For transmission efficiency, however, any sizable video data transfers between a transmitter and receiver should be performed in the YUV color format to take advantage of the compression schemes described below.

15

Luma Compression

In each data packet containing a video block, there are (source_w * source_h) luma values -- one for each pixel. If the luma encoding field

20 indicates that no encoding is being performed, the luma values are unsigned eight-bit values. If, however, luma encoding is indicated in the luma encoding field, the luma values are encoded to achieve a compression ratio of 2:1. In an embodiment of the invention, the luma value "Y" is compressed using a quantized differential coding (QDC) scheme described below. In other

25 embodiments, other compression schemes may be specified in the luma encoding field.

The luma compression herein described is based on the premise that luma values do not tend to vary significantly from one pixel to another. It is therefore possible to transmit the difference value between luma values for consecutive pixels rather than the luma values themselves. Further, the 5 luma difference values can be satisfactorily quantized to one of sixteen quantization levels, each of which is identified by a four-bit code word. The quantization is non-linear, with more quantization levels near zero where luma differences between consecutive pixels are more likely to occur.

10 In one embodiment, the luma difference quantization is performed according to the following table:

	<u>Difference Range</u>	<u>Code (Binary)</u>	<u>Quantized Difference Level</u>
15	-255 to -91	0 (0000)	-100
	-90 to -71	1 (0001)	-80
	-70 to -51	2 (0010)	-60
	-50 to -31	3 (0011)	-40
	-30 to -16	4 (0100)	-20
	-15 to -8	5 (0101)	-10
20	-7 to -3	6 (0110)	-4
	-2 to 0	7 (0111)	-1
	1 to 2	8 (1000)	1
	3 to 7	9 (1001)	4
	8 to 15	10 (1010)	10
	16 to 30	11 (1011)	20
25	31 to 50	12 (1100)	40
	51 to 70	13 (1101)	60
	71 to 90	14 (1110)	80
	91 to 255	15 (1111)	100

30

Figure 4A is a flow diagram describing how the luma compression process is performed in accordance with an embodiment of the invention. The scheme is based on a subtraction of a "last_value" from the current pixel luma value to generate the luma difference. "Last_value" is used to model 35 the luma value of the preceding pixel. To prevent divergence of the

compression and decompression processes, the "last_value" is modeled to account for the previous quantized luma difference rather than to match the actual luma value of the last pixel. The modeled "last_value" in the compression process therefore matches the corresponding modeled

5 "last_value" extracted in the decompression process.

Because the compression scheme is based on differences in luma values in rows of luma data, the first luma value in each row has no luma value with which to form a difference. To provide a starting point, in step

10 400, an initial "last_value" is assigned from the middle of the luma range. In step 401, the first luma value in the row of pixels is set as the current luma value. In step 402, the "last_value" is subtracted from the current luma value to generate a current luma difference value. The current luma difference is applied to a quantization function, in step 403, that outputs the

15 quantized difference code. In step 404, the difference code is placed in the video block data packet.

In step 405, the quantized difference level corresponding to the difference code is determined, and, in step 406, the "last_value" is updated by

20 incrementing by the quantized difference level. In step 407, "last_value" is clamped to prevent overflow. The clamping function is:

	<u>clamp(x)</u>	x
25	0	$x < 0$
	255	$x > 255$
	x	otherwise

In step 408, if there are more pixel luma values in the row, then

process flows to step 409 wherein the next luma value is set as the current

30 luma value. After step 409, the process returns to step 402. If there is no

further pixel luma value in the row at step 408, then, in step 410, a determination is made whether there are further rows to process in the video block. If there are further rows to compress, the next row is designated in step 411 and the process returns to step 400. If there are no further rows at step 5 410, the luma compression is completed for the current video block.

A luma decompression process is illustrated in the flow diagram of Figure 4B. In step 412, the "last_value" is set to the same midrange value as is done for the beginning of a row in the compression scheme. In step 413, 10 the first luma difference code is set as the current luma difference code. The quantized difference value is determined from the current luma difference code in step 414. In step 415, the "last_value" is incremented by the quantized difference value. In step 416, "last_value" is clamped to prevent overflow. In step 417, "last_value," now representing the decompressed current luma 15 value, is written to a buffer.

If, in step 418, there are further luma difference codes in the current row of the video block, the next difference code is set as the current luma difference code in step 419, and the process returns to step 414. If, in step 418, 20 there are no further luma difference codes in the current row, the process continues to step 420. In step 420, if there are no further rows in the video block, decompression is complete for the current video block. If, in step 420, there are further rows of luma difference codes, the next row of luma difference codes is set as the current row in step 421, and the process returns 25 to step 412.

Chroma Compression

The human eye is less sensitive to chroma information than to luma information, particularly in a spatial sense. For example, if, in generating an 5 image, some of the chroma information is spread beyond the actual edges of an object in the image, the human eye will typically pick up on the edge queues provided by the luma information and overlook the inaccuracies in the spatial location of the chroma information. For this reason, some latitude can be taken with the manner in which chroma information is 10 provided. Specifically, subsampling may be performed without significantly degrading visual quality. Subsampling may consist of sampling a single chroma value from

In accordance with an embodiment of the invention, the amount of 15 chroma information, and hence the amount of chroma compression, is specified by the `chroma_sub_X` and `chroma_sub_Y` fields in the video block data packet. If the values for both of those fields are zero, then there is no chroma information and the video block is monochrome, i.e., luma only. One possible specification for chroma subsampling is:

20

- 0 - No chroma values; monochrome image
- 1 - Subsample by one (i.e., no subsampling)
- 2 - Subsample by two
- 3 - Subsample by four

25

Further subsample arrangements may be provided by extending the above specification. `Chroma_sub_X` and `chroma_sub_Y` independently specify subsampling along respective axes. Several subsampling arrangements achieved by different combinations of `chroma_sub_X` and `chroma_sub_Y`, as 30 defined above, are:

	<u>chroma_sub_X</u>	<u>chroma_sub_Y</u>	<u>one chroma value per</u>	<u>compression</u>
5	0	0	0, no chroma data	--
	0	1	not permitted	--
	1	0	not permitted	--
	1	1	pixel	1:1
10	2	1	1 x 2 pixel array	2:1
	1	2	2 x 1 pixel array	2:1
	3	1	1 x 4 pixel array	4:1
	1	3	4 x 1 pixel array	4:1
15	3	2	2 x 4 pixel array	8:1
	2	3	4 x 2 pixel array	8:1
	3	3	4 x 4 pixel array	16:1

15 Subsampling may be performed when packing data into a video block data packet by taking data only at the specified intervals in the specified directions. For example, for (chroma_sub_X, chroma_sub_Y) = (3, 2), chroma data would be taken at every fourth pixel along each row, and every other row would be skipped. Other schemes may be used to select a single 20 pixel from the subsampling matrix, such as pseudo-random assignments. Further, the chroma values from each pixel in a subsampling matrix may be used to calculate a single set of average chroma values (U, V) for each subsampling matrix .

25 Subsampling is performed as the video block data packet is being packed and may occur before or after luma compression as luma and chroma compression are substantially independent. When the video block data packet reaches the receiver, the chroma subsamples are upsampled prior to being converted to RGB. Upsampling may be accomplished by taking the 30 subsampled chroma information and duplicating the chroma values for each pixel in the associated subsampling matrix.

Figure 5 illustrates subsampling and upsampling processes carried out on an 8 x 8 array of pixels with subsampling performed in 2 x 4 matrices (chroma_sub_X = 2, chroma_sub_Y = 3). 8 x 8 pixel array 500 represents the original video data prior to subsampling. 4 x 2 pixel array 501 represents the 5 video data after subsampling by the transmitter, and includes the data that would be transmitted to the receiver. 8 x 8 pixel array 502 represents the video data after upsampling at the receiver.

The subsampling matrices are identified in array 500 as those pixel cells 10 having the same index number. For example, all of the pixel cells containing a "1" are in the same subsampling matrix. 4 x 2 array 501 contains the subsampled data from array 500. The chroma values associated with those pixels with index "1" are averaged into chroma average value A1 (A1 comprises an averaged U value and an averaged V value) placed into the first 15 cell of array 501. Similarly, the chroma values for those pixels with index "2" are averaged into chroma average value A2 and placed into the second location in array 501. The other subsampling matrices indexed as "3"- "8" are averaged similarly. The compression ratio seen between array 500 and array 501 is 8:1.

20

Array 501 is upsampled into array 502 by placing the averaged chroma values A1-A8 into the positions corresponding to the respective original subsampling matrices. For example, averaged chroma value A1 is placed into each of the pixels in the upper left corner of 502 shown as containing "A1." 25 The insensitivity of the human eye to spatial errors in chroma information allows the averaged chroma values to provide satisfactory viewing results.

Upscaling And Downscaling Of Video Data

In an embodiment of the invention, the pixel array size of the source video block may differ from the size of the destination rectangle on the receiver's display. This size variation allows for a receiver with a large display to "blow up" or upscale a small video scene to make better use of the display resources. For example, a receiver may wish to upscale a 640 x 480 video stream to fill a 1024 x 1024 area on a large display device. Also, a receiver may have a smaller display than the size of a video stream. For this case, the video stream should be scaled down to be fully visible on the small display.

In accordance with an embodiment of the invention, upscaling is performed by the receiver, whereas downscaling is performed by the transmitter. One reason for this segregation of scaling duties is that scaled down video data requires lower network bandwidth to transmit. By downscaling video data on its own, the transmitter avoids sending video data that would be later discarded by the receiver. This also permits some simplification of the receiver in that resources, such as software code for downscaling video data, are not needed at the receiver.

Upscaling typically involves duplication of video data. It would be inefficient to send duplicated video data over a network. Therefore, the receiver performs all upscaling operations after receipt of the video data in its smaller form. Upscaling of video data is supported in the fields associated with the video data packet. Specifically, the video protocol provides separate fields for specifying the video source pixel array size and the destination

display rectangle size. The amount of horizontal scaling is (width/source_w), and the amount of vertical scaling is (height/source_h).

Upscaling is performed after the video data has been decompressed and
5 transformed into RGB format, though in certain embodiments upscaling may precede, or be combined with, the decompression steps. The receiver expands the video data vertically, horizontally or both as need to make the video data fill the designated display rectangle. Expanding video data may be performed as simply as doubling pixel values, but more advanced image filtering
10 techniques may be used to affect re-sampling of the image for better display quality.

Video Data Process Implementing Protocol

15 Figure 6A is a flow diagram illustrating how a transmitter processes video data in accordance with an embodiment of the invention. In step 600, the transmitter acquires video data for transmission to a receiver. The video data may be acquired by any mechanism, such as capture of a video signal using a hardware capture board, generation of video data by a video service, 20 or input of video data from a video input device such as a video camera.

In step 601, if necessary, the video data is decompressed or converted into YUV color format in accordance with the established protocol. In step 602, the transmitter downscales the video data if the transmitter determines 25 that downscaling is needed. The luma values of the YUV video data are compressed, in step 603, using the quantized differential coding (QDC) scheme described herein, and loaded into a data packet. In step 604, the transmitter subsamples the chroma values of the YUV video data and loads

the subsampled chroma values into the data packet. The completed data packet containing the video data for a video block is sent to a receiver in step 605. After transmitting the data packet, the process returns to step 600.

5 Figure 6B is a flow diagram illustrating how a receiver processes video data in accordance with an embodiment of the invention. In block 606, the receiver receives a compressed/subsampled YUV video block data packet from the transmitter. The receiver decompresses the luma values of the data packet in step 607, and upsamples the chroma values in step 608. With full 10 YUV video data, the receiver performs a color transformation to convert the YUV video data to RGB data. If the destination display rectangle noted in the data packet header is larger than the source video data, the receiver performs any necessary upscaling to fill the designated display rectangle with the source video data. In step 611, the video data is loaded into a video buffer for display 15 on a the receiver's display device. After loading the video data into the video buffer, the process returns to step 606.

Embodiment of Computer Execution Environment (Hardware)

20 An embodiment of the invention can be implemented as computer software in the form of computer readable code executed on a general purpose computers such as computer 700 illustrated in Figure 7, or in the form of bytecode class files executable within a Java™ runtime environment running on such a computer. A keyboard 710 and mouse 711 are coupled to a 25 bi-directional system bus 718. The keyboard and mouse are for introducing user input to the computer system and communicating that user input to processor 713. Other suitable input devices may be used in addition to, or in place of, the mouse 711 and keyboard 710. I/O (input/output) unit 719

coupled to bi-directional system bus 718 represents such I/O elements as a printer, A/V (audio/video) I/O, etc.

Computer 700 includes a video memory 714, main memory 715 and 5 mass storage 712, all coupled to bi-directional system bus 718 along with keyboard 710, mouse 711 and processor 713. The mass storage 712 may include both fixed and removable media, such as magnetic, optical or magnetic optical storage systems or any other available mass storage technology. Bus 718 may contain, for example, thirty-two address lines for 10 addressing video memory 714 or main memory 715. The system bus 718 also includes, for example, a 32-bit data bus for transferring data between and among the components, such as processor 713, main memory 715, video memory 714 and mass storage 712. Alternatively, multiplex data/address lines may be used instead of separate data and address lines.

15

In one embodiment of the invention, the processor 713 is a microprocessor manufactured by Motorola, such as the 680X0 processor or a microprocessor manufactured by Intel, such as the 80X86, or Pentium processor, or a SPARC™ microprocessor from Sun Microsystems™, Inc.

20 However, any other suitable microprocessor or microcomputer may be utilized. Main memory 715 is comprised of dynamic random access memory (DRAM). Video memory 714 is a dual-ported video random access memory. One port of the video memory 714 is coupled to video amplifier 716. The video amplifier 716 is used to drive the cathode ray tube (CRT) raster monitor 25 717. Video amplifier 716 is well known in the art and may be implemented by any suitable apparatus. This circuitry converts pixel data stored in video memory 714 to a raster signal suitable for use by monitor 717. Monitor 717 is a type of monitor suitable for displaying graphic images. Alternatively, the

video memory could be used to drive a flat panel or liquid crystal display (LCD), or any other suitable data presentation device.

Computer 700 may also include a communication interface 720 coupled to bus 718. Communication interface 720 provides a two-way data communication coupling via a network link 721 to a local network 722. For example, if communication interface 720 is an integrated services digital network (ISDN) card or a modem, communication interface 720 provides a data communication connection to the corresponding type of telephone line, which comprises part of network link 721. If communication interface 720 is a local area network (LAN) card, communication interface 720 provides a data communication connection via network link 721 to a compatible LAN. Communication interface 720 could also be a cable modem or wireless interface. In any such implementation, communication interface 720 sends and receives electrical, electromagnetic or optical signals which carry digital data streams representing various types of information.

Network link 721 typically provides data communication through one or more networks to other data devices. For example, network link 721 may provide a connection through local network 722 to local server computer 723 or to data equipment operated by an Internet Service Provider (ISP) 724. ISP 724 in turn provides data communication services through the world wide packet data communication network now commonly referred to as the "Internet" 725. Local network 722 and Internet 725 both use electrical, electromagnetic or optical signals which carry digital data streams. The signals through the various networks and the signals on network link 721 and through communication interface 720, which carry the digital data to and

from computer 700, are exemplary forms of carrier waves transporting the information.

Computer 700 can send messages and receive data, including program
5 code, through the network(s), network link 721, and communication
interface 720. In the Internet example, remote server computer 726 might
transmit a requested code for an application program through Internet 725,
ISP 724, local network 722 and communication interface 720.

10 The received code may be executed by processor 713 as it is received,
and/or stored in mass storage 712, or other non-volatile storage for later
execution. In this manner, computer 700 may obtain application code in the
form of a carrier wave.

15 Application code may be embodied in any form of computer program
product. A computer program product comprises a medium configured to
store or transport computer readable code or data, or in which computer
readable code or data may be embedded. Some examples of computer
program products are CD-ROM disks, ROM cards, floppy disks, magnetic
20 tapes, computer hard drives, servers on a network, and carrier waves.

Human Interface Device Computer System

The invention has application to computer systems where the data is
25 provided through a network. The network can be a local area network, a
wide area network, the internet, world wide web, or any other suitable
network configuration. One embodiment of the invention is used in

computer system configuration referred to herein as a human interface device computer system.

In this system the functionality of the system is partitioned between a
5 display and input device, and data sources or services. The display and input device is a human interface device (HID). The partitioning of this system is such that state and computation functions have been removed from the HID and reside on data sources or services. In one embodiment of the invention, one or more services communicate with one or more HIDs through some
10 interconnect fabric, such as a network. An example of such a system is illustrated in Figure 8. Referring to Figure 8, the system consists of computational service providers 800 communicating data through interconnect fabric 801 to HIDs 802.

15 Computational Service Providers - In the HID system, the computational power and state maintenance is found in the service providers, or services. The services are not tied to a specific computer, but may be distributed over one or more traditional desktop systems such as described in connection with Figure 7, or with traditional servers. One
20 computer may have one or more services, or a service may be implemented by one or more computers. The service provides computation, state, and data to the HIDs and the service is under the control of a common authority or manager. In Figure 8, the services are found on computers 810, 811, 812, 813, and 814. In an embodiment of the invention, any of computers 810-814 could
25 be implemented as a transmitter.

Examples of services include X11/Unix services, archived video services, Windows NT service, JavaTM program execution service, and others.

A service herein is a process that provides output data and responds to user requests and input.

Interconnection Fabric - The interconnection fabric is any of multiple 5 suitable communication paths for carrying data between the services and the HIDs. In one embodiment the interconnect fabric is a local area network implemented as an Ethernet network. Any other local network may also be utilized. The invention also contemplates the use of wide area networks, the internet, the world wide web, and others. The interconnect fabric may be 10 implemented with a physical medium such as a wire or fiber optic cable, or it may be implemented in a wireless environment.

HIDs - The HID is the means by which users access the computational services provided by the services. Figure 8 illustrates HIDs 821, 822, and 823. 15 A HID consists of a display 826, a keyboard 824, mouse 825, and audio speakers 827. The HID includes the electronics need to interface these devices to the interconnection fabric and to transmit to and receive data from the services. In an embodiment of the invention, an HID is implemented as a receiver.

20 A block diagram of the HID is illustrated in Figure 9. The components of the HID are coupled internally to a PCI bus 912. A network control block 902 communicates to the interconnect fabric, such as an ethernet, through line 914. An audio codec 903 receives audio data on interface 916 and is coupled to block 902. USB data communication is provided on lines 913 to 25 USB controller 901.

An embedded processor 904 may be, for example, a Sparc2ep with coupled flash memory 905 and DRAM 906. The USB controller 901, network

controller 902 and embedded processor 904 are all coupled to the PCI bus 912. Also coupled to the PCI 912 is the video controller 909. The video controller 909 may be for example, and ATI RagePro+ frame buffer controller that provides SVGA output on line 915. NTSC data is provided in and out of the 5 video controller through video decoder 910 and video encoder 911 respectively. A smartcard interface 908 may also be coupled to the video controller 909.

10 The computer systems described above are for purposes of example only. An embodiment of the invention may be implemented in any type of computer system or programming or processing environment.

15 Thus, a method and apparatus of supporting a video protocol in a network environment have been described in conjunction with one or more specific embodiments. The invention is defined by the claims and their full scope of equivalents.

CLAIMS

1. An apparatus comprising:
one or more video sources providing video data in accordance with a
5 plurality of video protocols;
a receiver configured to support a single video protocol;
a transmitter coupled to said receiver over a network, said transmitter
configured to convert said video data from said one or more video sources
into converted video data that conforms to said single video protocol.

10

2. The apparatus of claim 1, wherein:
said transmitter is configured to perform downscaling of said
converted video data; and
said receiver is configured to perform upscaling of said converted
15 video data.

3. The apparatus of claim 1, wherein said transmitter is configured
to transmit said converted video data to said receiver in a plurality of packets.

20

4. The apparatus of claim 3, wherein said receiver is configured to
process each of said plurality of packets independently.

5. The apparatus of claim 3, wherein said transmitter is configured
to pseudo-randomly transmit said plurality of packets.

25

6. The apparatus of claim 3, wherein each of said plurality of
packets comprises a sequence number.

7. The apparatus of claim 6, wherein said receiver is configured to communicate skipped sequence numbers to said transmitter.

8. The apparatus of claim 7, wherein said transmitter is configured
5 to retransmit one or more packets corresponding to said skipped sequence numbers.

9. The apparatus of claim 7, wherein said transmitter is configured to transmit one or more updated packets corresponding to said skipped
10 sequence numbers.

10. The apparatus of claim 3, wherein each of said plurality of packets further comprises one or more coordinate fields specifying one or more coordinates of a first pixel of a block of video data.

15

11. The apparatus of claim 10, wherein each of said plurality of packets further comprises:

a width field specifying the width of a destination rectangle at said receiver; and

20 a height field specifying the height of said destination rectangle.

12. The apparatus of claim 10, wherein each of said plurality of packets further comprises:

a width field specifying the width of said block of video data; and

25 a height field specifying the height of said block of video data.

13. The apparatus of claim 3, wherein each of said plurality of packets comprises a luma encoding field specifying a luma encoding scheme from a plurality of luma encoding schemes.

5 14. The apparatus of claim 3, wherein each of said plurality of packets comprises one or more chroma subsampling fields specifying one or more subsampling values from a plurality of subsampling values.

10 15. In a computer network, a method for transmitting video data comprising:

a transmitter receiving video data from one or more sources; said transmitter converting said video data into a plurality of packets in accordance with a protocol; said transmitter transmitting said plurality of packets to a receiver; and 15 said receiver displaying each of said plurality of packets independently.

20 16. The method of claim 15, wherein converting said video data further comprises applying quantized differential encoding to a plurality of luma values of said video data.

17. The method of claim 15, wherein converting said video data further comprises subsampling one or more chroma values of said video data.

25 18. The method of claim 15, wherein converting said video data further comprises downscaling said video data.

19. The method of claim 15, wherein displaying comprises upscaling said video data.

20. The method of claim 15, wherein converting said video data
5 further comprises associating a sequence number with each of said plurality of packets.

21. The method of claim 20, further comprising:
said receiver communicating one or more skipped sequence numbers
10 to said transmitter; and
said transmitter transmitting one or more packets of video data corresponding to said skipped sequence numbers.

22. The method of claim 15, wherein converting said video data
15 further comprises writing one or more coordinate fields in each of said plurality of packets, said one or more coordinate fields specifying one or more coordinates of a pixel of a video block.

23. The method of claim 15, wherein converting said video data
20 further comprises:
writing a width field in each of said plurality of packets, said width field specifying the width of a destination rectangle of said receiver; and
writing a height field in each of said plurality of packets, said height field specifying the height of said destination rectangle.

24. The method of claim 15, wherein converting said video data further comprises:

writing a width field in each of said plurality of packets, said width field specifying the width of a block of video data stored in a respective packet;

5 and

writing a height field in each of said plurality of packets, said height field specifying the height of said block of video data.

25. The method of claim 15, wherein converting said video data 10 further comprises writing a luma encoding field in each of said plurality of packets.

26. The method of claim 15, wherein converting said video data further comprises writing one or more chroma subsample values in each of 15 said plurality of packets.

27. A computer program product comprising:
a computer usable medium having computer readable code embodied therein for transmitting video data in a computer network, said computer 20 program product comprising:

computer readable code configured to cause a transmitter to perform the steps of:

receiving video data from one or more sources;
converting said video data into a plurality of packets in
25 accordance with a protocol; and
transmitting said plurality of packets to a receiver;
computer readable code configured to cause a receiver to display each
of said plurality of packets independently.

28. The computer program product of claim 27, wherein converting said video data further comprises applying quantized differential encoding to a plurality of luma values of said video data.

5

29. The computer program product of claim 27, wherein converting said video data further comprises subsampling one or more chroma values of said video data.

10

30. The computer program product of claim 27, wherein converting said video data further comprises downscaling said video data.

15

31. The computer program product of claim 27, further comprising computer readable code configured to cause said receiver to upscale said video data.

32. The computer program product of claim 27, wherein converting said video data further comprises associating a sequence number with each of said plurality of packets.

20

33. The computer program product of claim 32, further comprising: computer readable code configured to cause said receiver to communicate one or more skipped sequence numbers to said transmitter; and

25

computer readable code configured to cause said transmitter to transmit one or more packets of video data corresponding to said skipped sequence numbers.

34. The computer program product of claim 27, wherein converting said video data further comprises writing one or more coordinate fields in each of said plurality of packets, said one or more coordinate fields specifying one or more coordinates of a pixel of a video block.

5

35. The computer program product of claim 27, wherein converting said video data further comprises:

writing a width field in each of said plurality of packets, said width field specifying the width of a destination rectangle of said receiver; and

10 writing a height field in each of said plurality of packets, said height field specifying the height of said destination rectangle.

36. The computer program product of claim 27, wherein converting said video data further comprises:

15 writing a width field in each of said plurality of packets, said width field specifying the width of a block of video data stored in a respective packet; and

writing a height field in each of said plurality of packets, said height field specifying the height of said block of video data.

20

37. The computer program product of claim 27, wherein converting said video data further comprises writing a luma encoding field in each of said plurality of packets.

25

38. The computer program product of claim 27, wherein converting said video data further comprises writing one or more chroma subsample values in each of said plurality of packets.

39. An apparatus comprising:
 - means for receiving video data from one or more sources;
 - means for converting said video data into a plurality of packets in accordance with a protocol;
 - 5 means for transmitting said plurality of packets to a receiver; and
 - means at said receiver for displaying each of said plurality of packets independently.

1/9

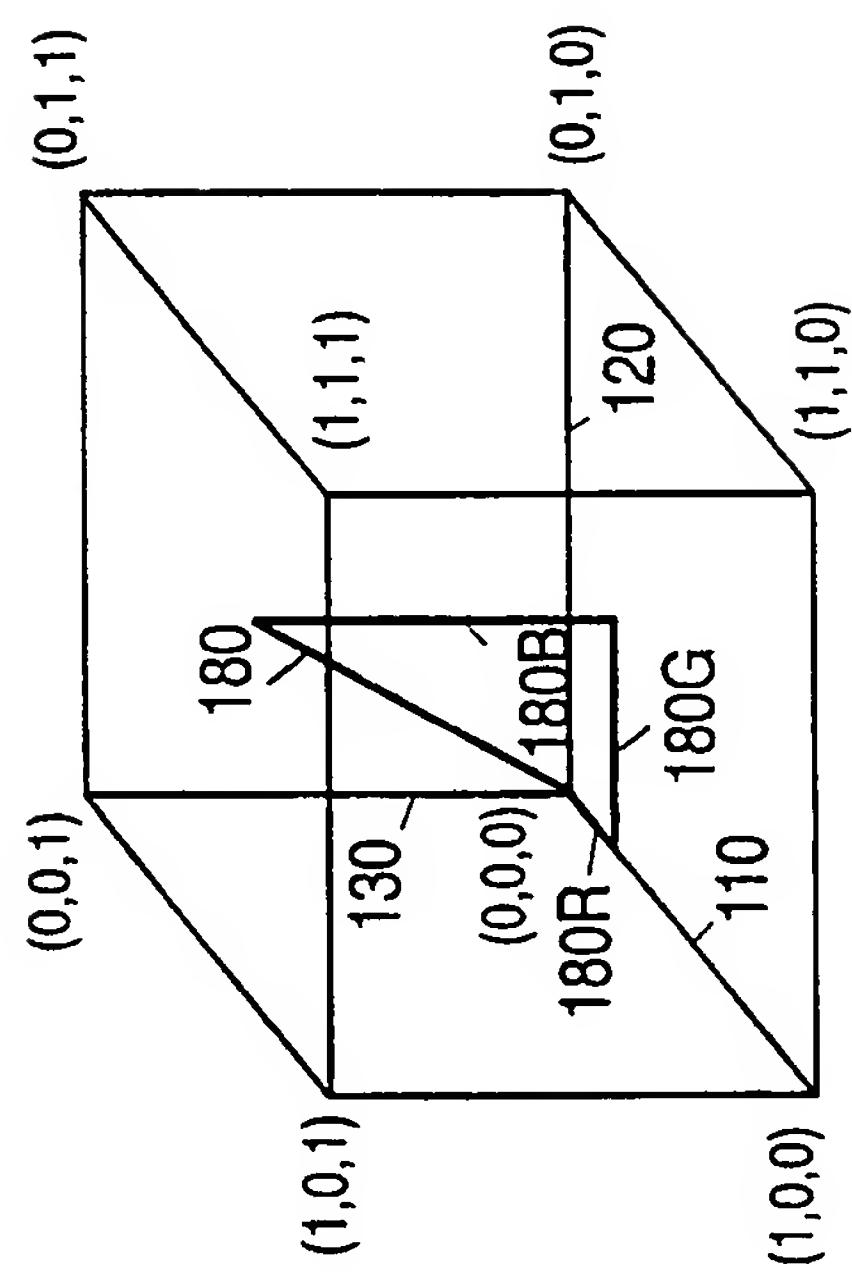


FIG. 1

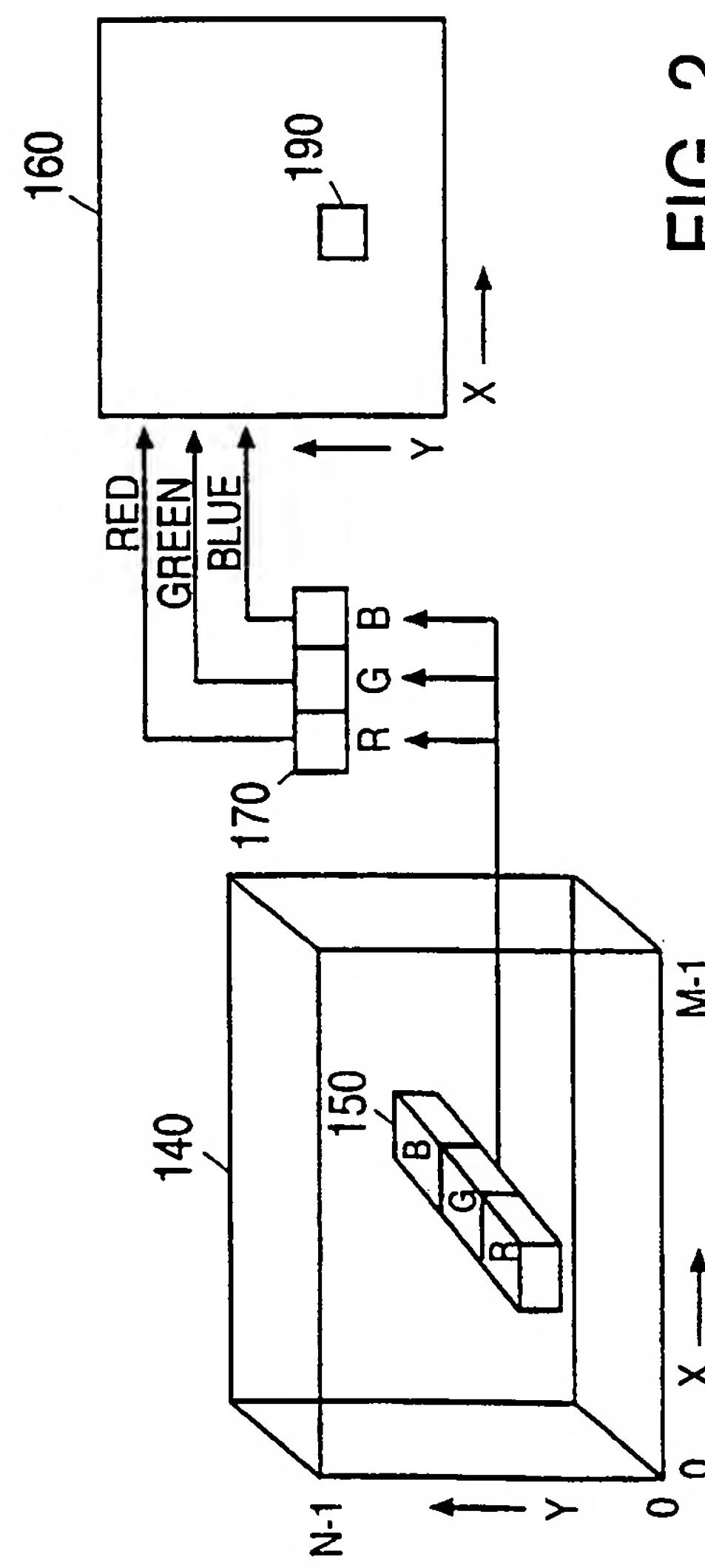


FIG. 2

2/9

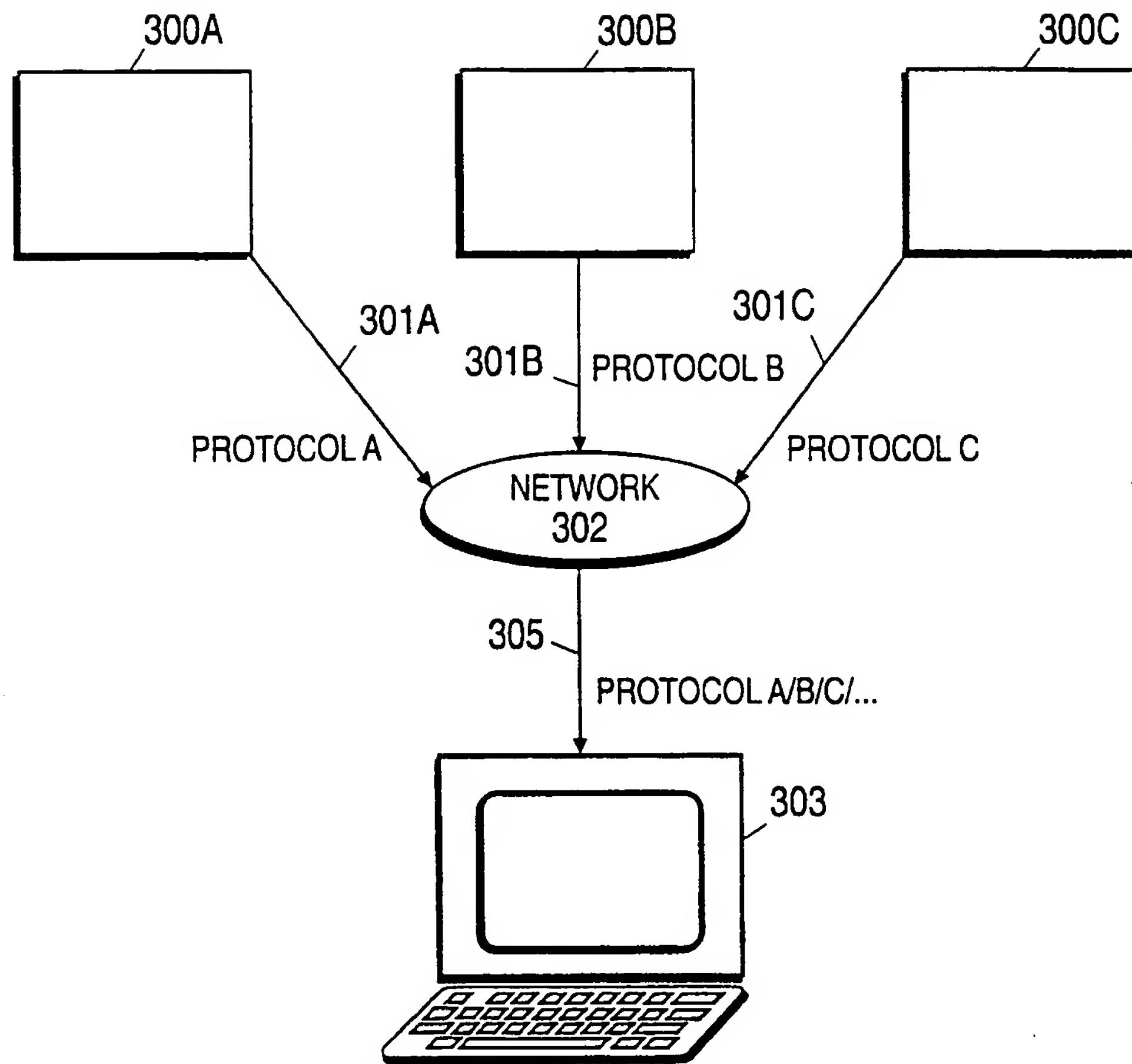


FIG. 3

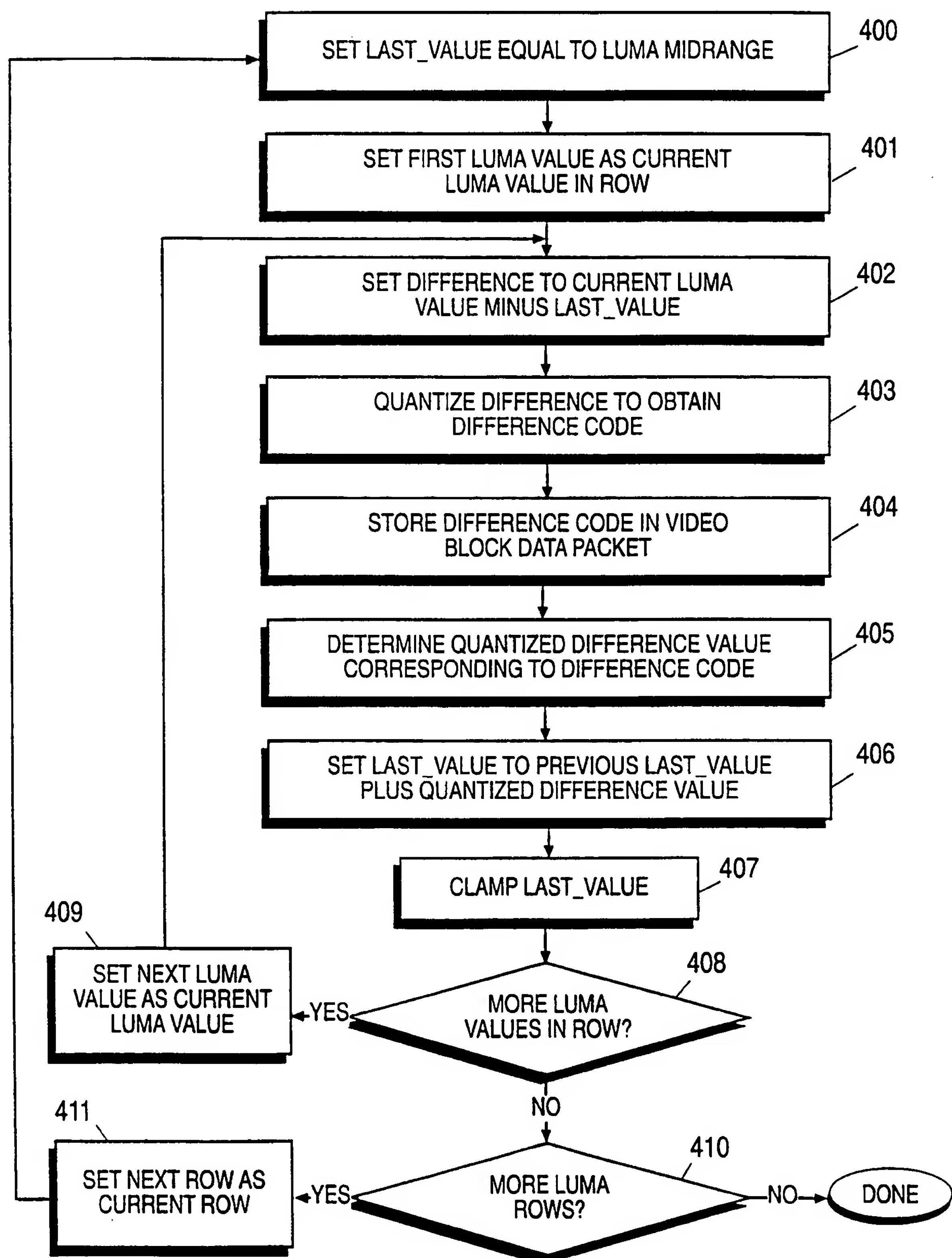


FIG. 4A

4/9

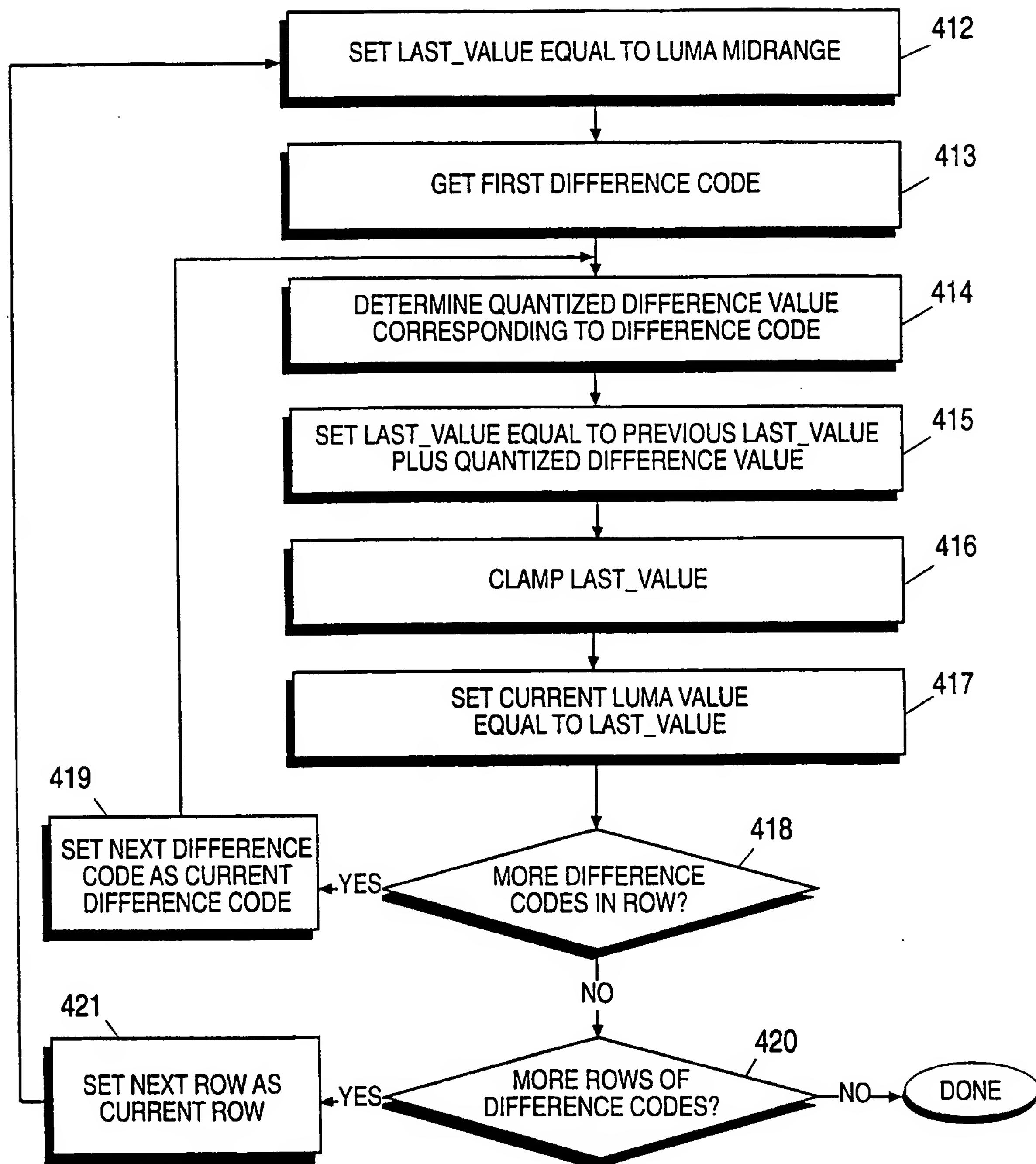


FIG. 4B

5/9

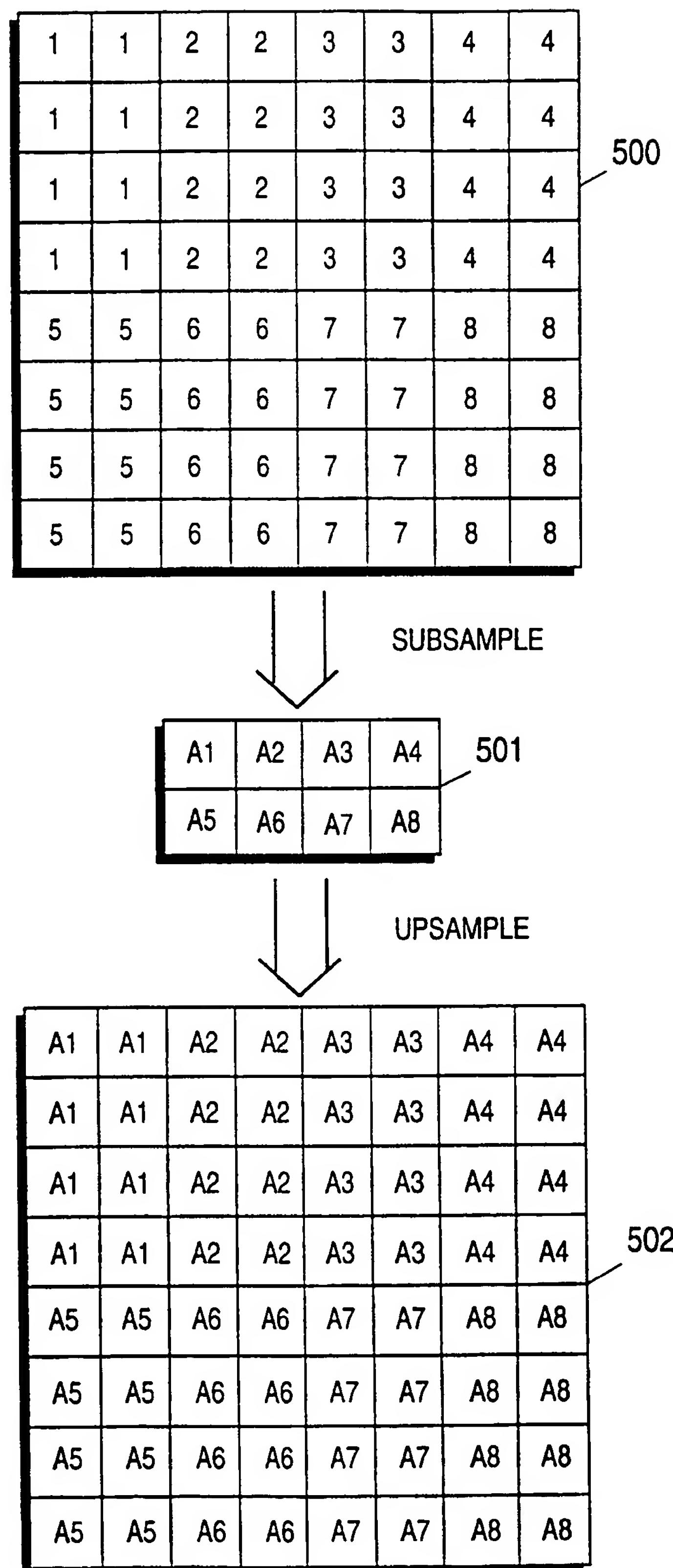


FIG. 5

6/9

FIG. 6A

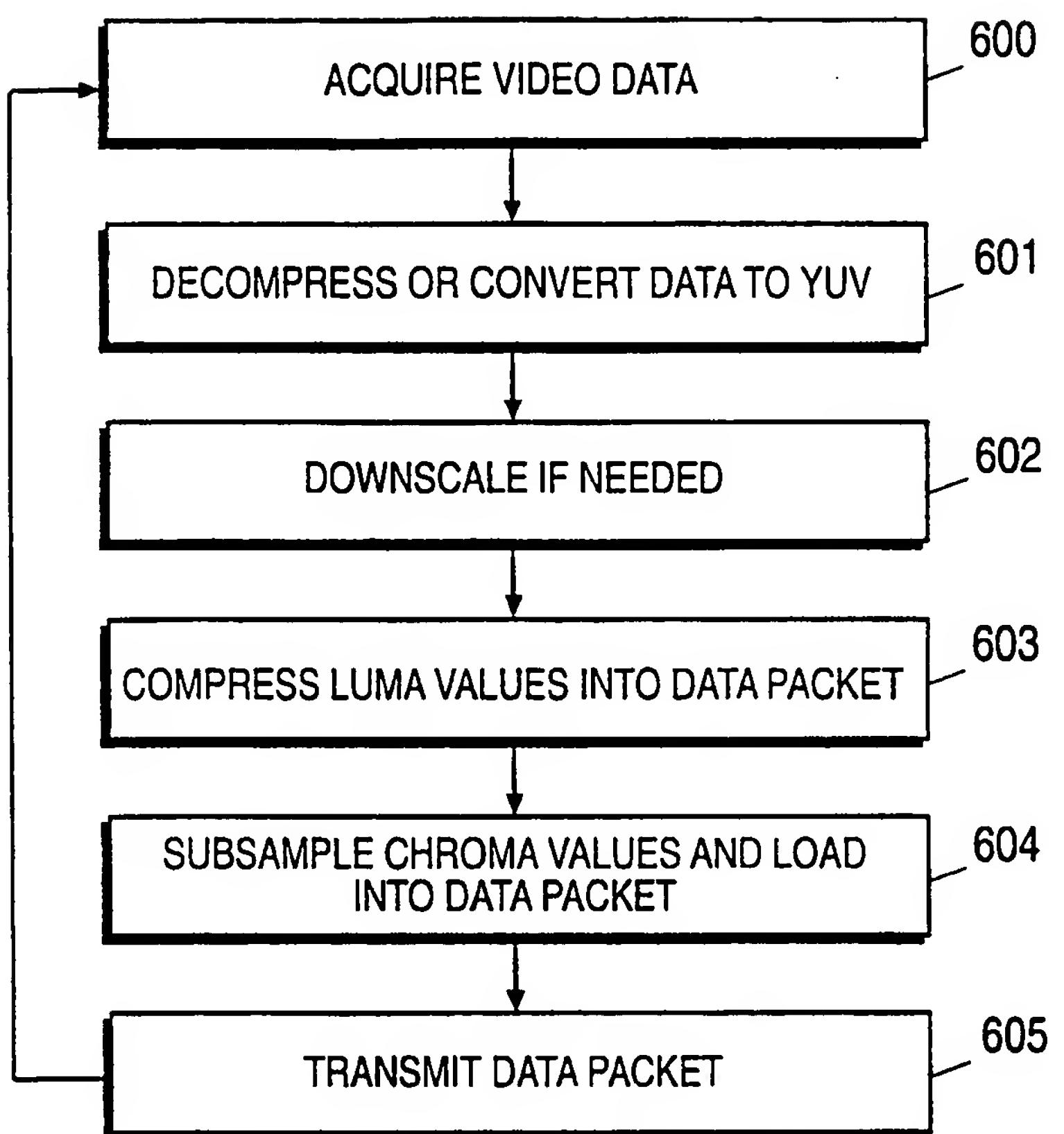
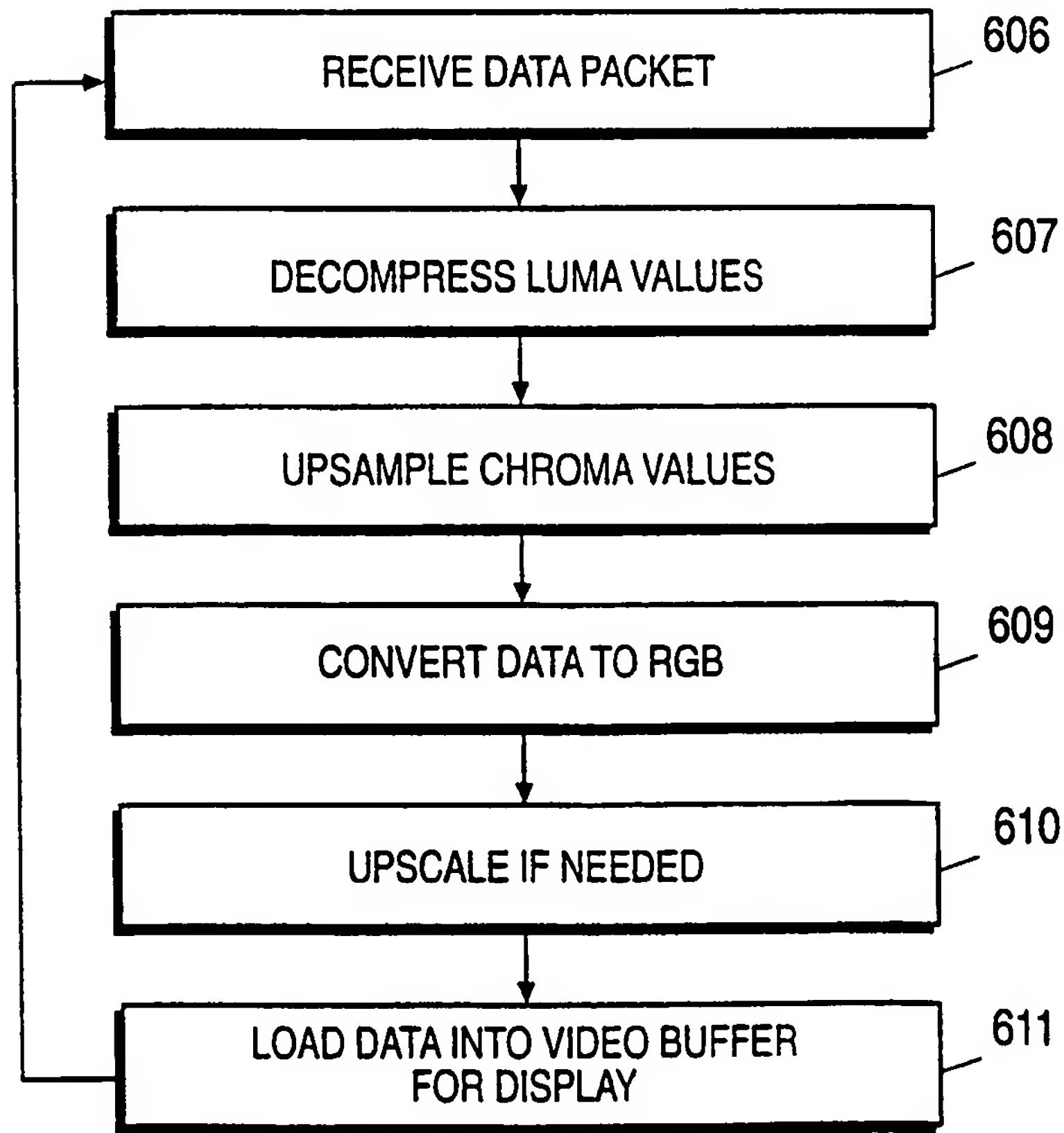


FIG. 6B



7/9

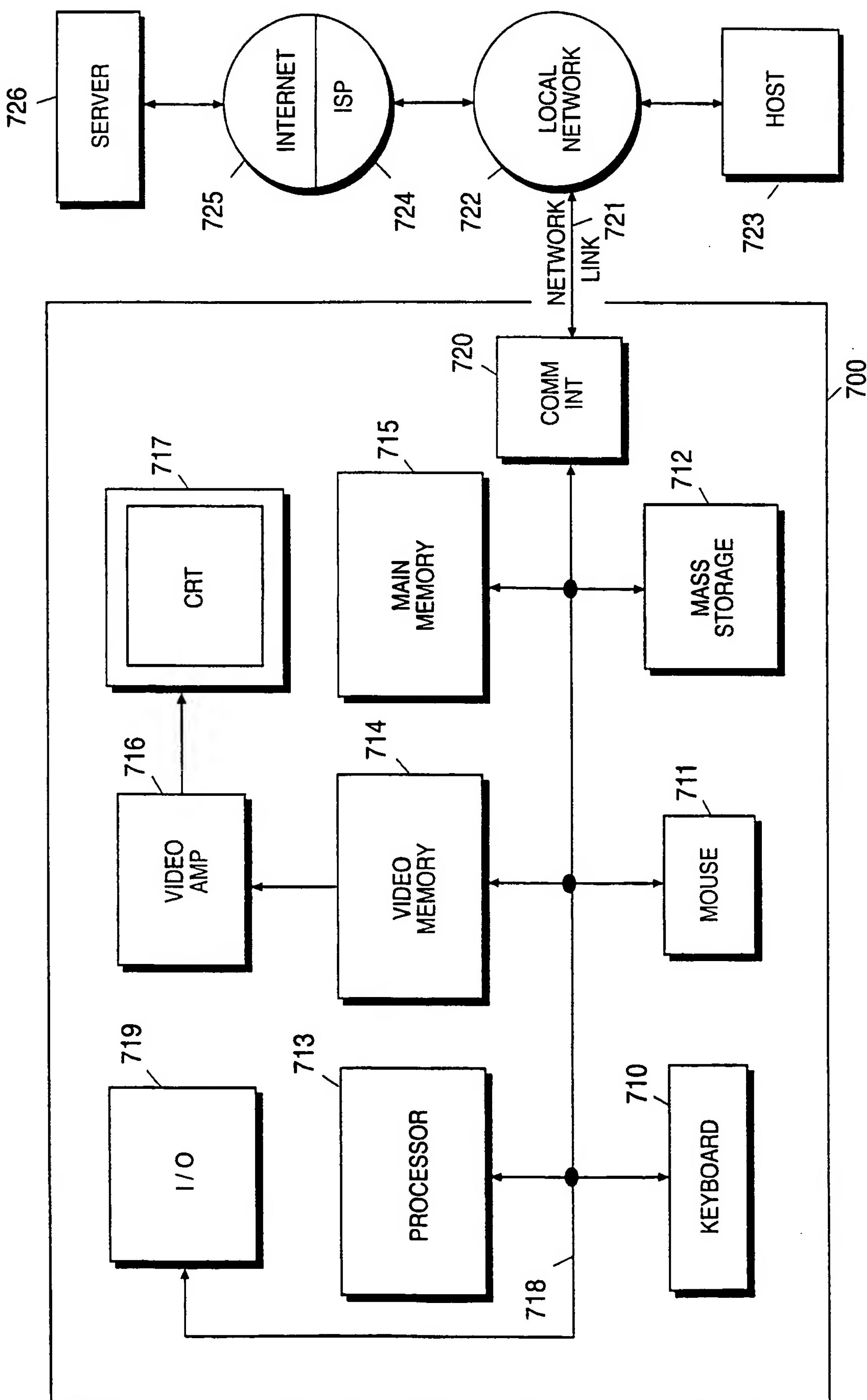


FIG. 7

8/9

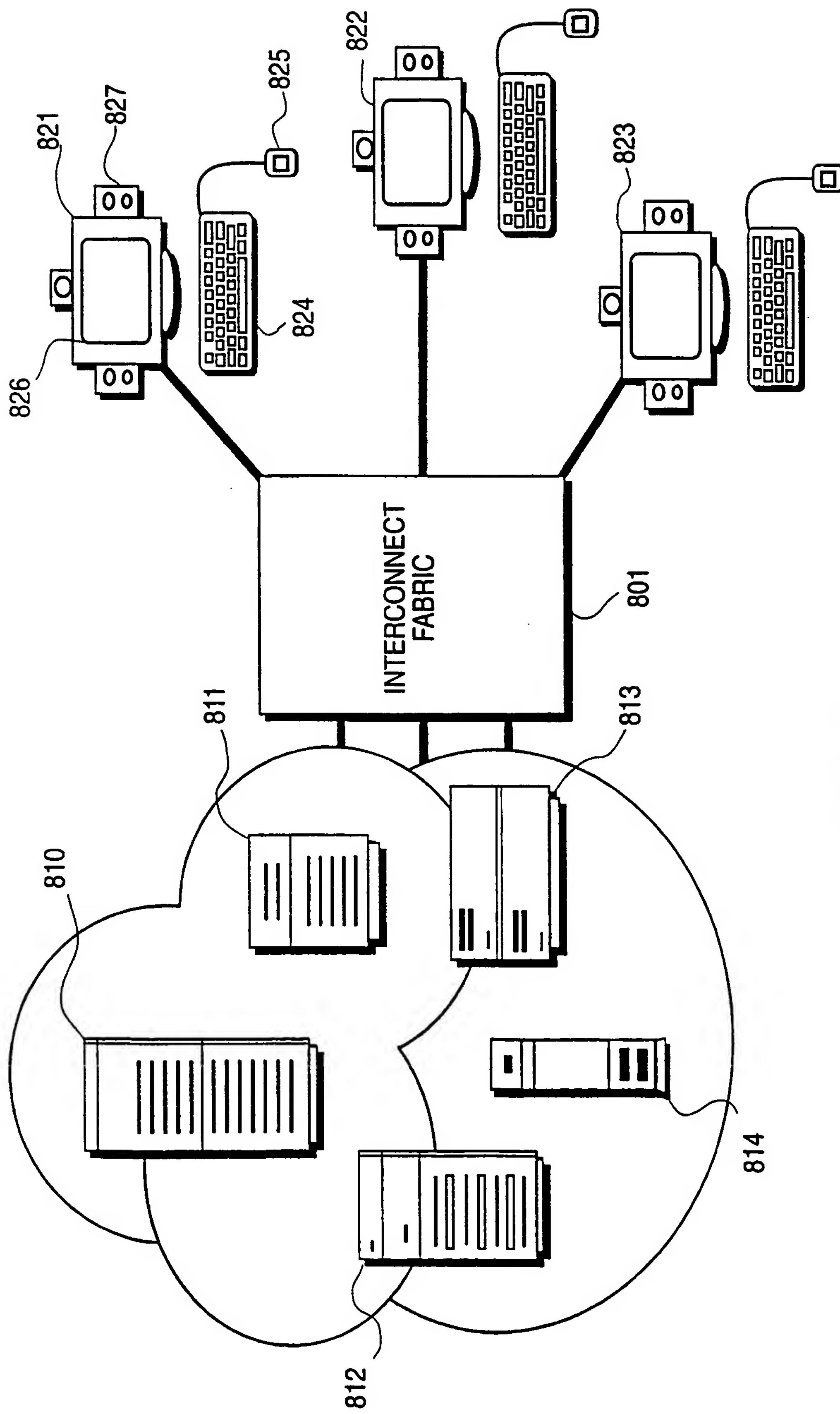
COMPUTATIONAL SERVICE PROVIDERS 800
HUMAN INTERFACE DEVICES 802

FIG. 8

9/9

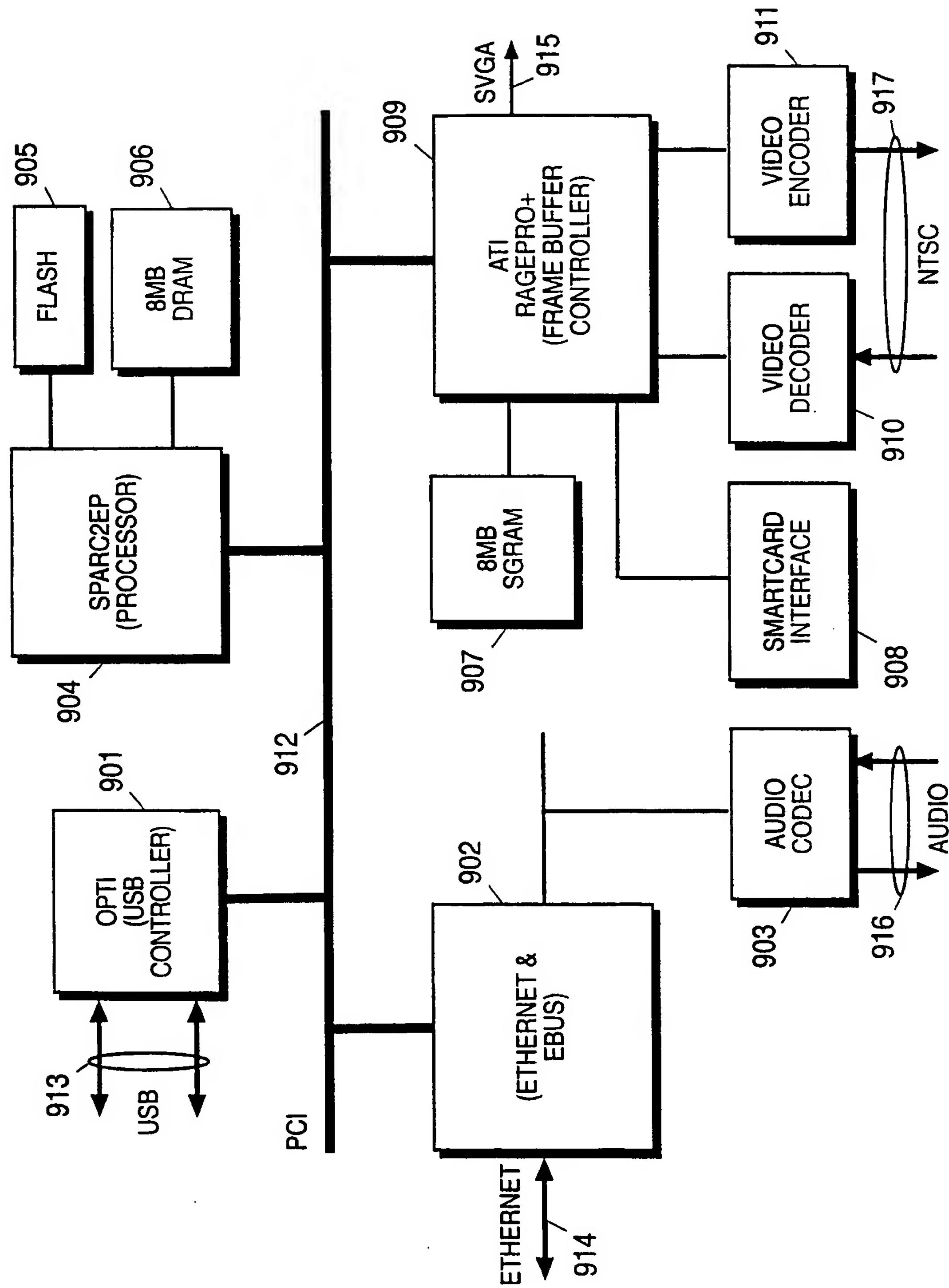


FIG. 9